# TECHNICAL RESEARCH REPORT

Multi-time Scale Markov Decision Processes

*by Hyeong Soo Chang, Pedram Fard,*
*Steven I. Marcus and Mark Shayman*

**TR 2002-6**

# Report Documentation Page

*Form Approved*
*OMB No. 0704-0188*

| | | |
|---|---|---|
| 1. REPORT DATE<br>**2002** | 2. REPORT TYPE | 3. DATES COVERED<br>**-** |
| 4. TITLE AND SUBTITLE<br>**Multi-time Scale Markov Decision Processes** | | 5a. CONTRACT NUMBER |
| | | 5b. GRANT NUMBER |
| | | 5c. PROGRAM ELEMENT NUMBER |
| 6. AUTHOR(S) | | 5d. PROJECT NUMBER |
| | | 5e. TASK NUMBER |
| | | 5f. WORK UNIT NUMBER |
| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)<br>**Air Force Office of Scientific Research,875 North Randolph Street,Arlington,VA,22203-1768** | | 8. PERFORMING ORGANIZATION REPORT NUMBER |
| 9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) | | 10. SPONSOR/MONITOR’S ACRONYM(S) |
| | | 11. SPONSOR/MONITOR’S REPORT NUMBER(S) |
| 12. DISTRIBUTION/AVAILABILITY STATEMENT<br>**Approved for public release; distribution unlimited** | | |
| 13. SUPPLEMENTARY NOTES | | |
| 14. ABSTRACT<br>**see report** | | |
| 15. SUBJECT TERMS | | |

| 16. SECURITY CLASSIFICATION OF: | | | 17. LIMITATION OF ABSTRACT | 18. NUMBER OF PAGES | 19a. NAME OF RESPONSIBLE PERSON |
|---|---|---|---|---|---|
| a. REPORT | b. ABSTRACT | c. THIS PAGE | | | |
| **unclassified** | **unclassified** | **unclassified** | | **35** | |

# Multi-time Scale Markov Decision Processes

Hyeong Soo Chang, Pedram Fard, Steven I. Marcus[*] and Mark Shayman
Institute for Systems Research
University of Maryland, College Park, MD 20742
E-mail: {hyeong,pfard,marcus,shayman}@isr.umd.edu

February 24, 2002

**Abstract**

This paper proposes a simple analytical model called $M$ time-scale Markov Decision Process (MMDP) for hierarchically structured sequential decision making processes, where decisions in each level in the $M$-level hierarchy are made in $M$ different time-scales. In this model, the state space and the control space of each level in the hierarchy are non-overlapping with those of the other levels, respectively, and the hierarchy is structured in a "pyramid" sense such that a decision made at level $m$ (slower time-scale) state and/or the state will affect the evolutionary decision making process of the lower level $m+1$ (faster time-scale) until a new decision is made at the higher level but the lower level decisions themselves do not affect the higher level's transition dynamics. The performance produced by the lower level's decisions will affect the higher level's decisions. A hierarchical objective function is defined such that the finite-horizon value of following a (nonstationary) policy at the level $m+1$ over a decision epoch of the level $m$ plus an immediate reward at the level $m$ is the single step reward for the level $m$ decision making process. From this we define "multi-level optimal value function" and derive "multi-level optimality equation". We discuss how to solve MMDPs exactly or approximately and also study heuristic on-line methods to solve MMDPs. Finally, we give some example control problems that can be modeled as MMDPs.

**Keywords:** Markov decision process, multi-time scale, hierarchical control

[*]Correspondence author.

S. I. Marcus can be reached by telephone at +1 (301) 405-3683 or by fax at +1 (301) 314-9920.

This research was supported in part by the National Science Foundation under Grant DMI-9988867, by the Air Force Office of Scientific Research under Grant F496200110161, and by the Department of Defense under Contract MDA 90499C2521.

## 1 Introduction

Hierarchically structured control problems have been studied extensively in many contexts in various areas over the past years with certain types of models and assumptions. This is because many large-scale control problems that arise in real applications show multi-dimensionally interdependent organized behavior among subsystems that constitute the whole system as decision blocks. Two distinguished hierarchical structures studied in the literature are “multi-level structure”, where decision making algorithms in different levels operate in different time scales (see, e.g., [22]) and “multi-layer structure”, where algorithms are divided “spatially” and operate at the same time scale (see, e.g., [13]).

This paper focuses on the control problems with a particular multi-level structure — *hierarchically structured sequential decision making processes*, where decisions in each level in the hierarchy are made in different time-scales and the hierarchy is structured in a pyramid (bottom-up organization) sense. That is, decisions made in the higher level affect the decision making process of the lower level but the lower level decisions do not affect the higher level (state transition) dynamics even though the performance produced by the lower level decisions will affect the decisions that will be made by the higher level.

An usual approach to the multi-level structured problems is that a slow time-scale subsystem lays aside the details of a fast time-scale dynamics by “average” behavior and then solves its own optimization problem (see, e.g., [14] [6] or chapter 11 in [33]). This approach makes sense especially when the hierarchy in the system is structured in the pyramid sense. This pyramid-like structure was used in the perspective of “performability” and “dependability” in Trivedi *et al.*’s models [23] [15] [25] even though controls are not involved in the models. They proposed a hierarchical performability and dependability model, where the performance models (fast time-scale model) are solved to obtain performance measures, termed as *quasi-steady state* performance. These measures are used as reward rates which are assigned to states of the dependability model (slow time-scale model). The dependability model is then solved to obtain performability measures. The lower level is modeled by a continuous-time Markov chain and the upper level is modeled by a Markov reward process.

In this paper, we propose a simple analytical model that generalizes Trivedi *et al.*’s hierarchical model by incorporating controls into the model, which we refer to as Multi-time scale Markov Decision Process (MMDP). The model describes interactions between levels in a hierarchy in the pyramid sense. Each level is associated with distinct state and action spaces. That is, we assume that no two level share any state or control action. The upper level state and/or control induces the lower level MDP dynamics over a finite horizon of length corresponding to the decision epoch of the upper level. In other words, a particular pair of the upper level state and/or control will determine the lower level’s state transition function and reward function. Hierarchical objective functions are defined such that the (quasi-steady state) performance measure, the finite horizon value of following a given lower level policy, obtained from the lower level over the decision epoch of the upper level will affect the upper level decision making. From this we define "multi-level value function" and then drive "multi-level optimality equation" for infinite horizon discounted reward and average reward case, respectively. We study how to compute the optimal multi-level value function exactly and approximately. We present an approximation method suited for solving MMDPs and analyze its performance and discuss how to apply some previously published on-line solution schemes in the context of MMDPs.

In addition to inherently existing hierarchical and multi-time scale control structure in problems themselves that arise in many different contexts, our model is motivated by in particular the observation made in the networking literature recently. The network traffic shows fluctuations on multiple time-scales — scale invariant burstiness (see, e.g., [41]), and this characteristic in the network traffic has been well-studied by "long-range dependent" or "self-similar" model (see, e.g., [42]). However, there are several recent works that investigated the effects of such multi-time scaled behavior by certain relevant Markovian models that approximate the fluctuations in the traffic (see, e.g., [34] [37] [24] and references therein). The usual interests are in calculations of buffer overflow probability distribution but are not concerned with development of analytical multi-time scaled controls that incorporate given traffic models for such behaviors of the network traffic even though some non-Markovian model based approaches are available (see, e.g., [38] and [16]). For example, the slow time-scale ("call-level") relates to the arrival and departure process of video/voice calls and the fast time-scale ("packet-level") relates to the packet arrival process of calls during their "lifetimes". This different time-scaled dynamics causes fluctuations in the traffic at different time-scales and gives rise to a multi-time scaled queueing control problem[1] and we believe that we need to develop an analytical model to approach this kind of control problems.

This paper is organized as follows. We present a formal description of MMDPs and characterize optimal solutions for MMDPs in Section 2 and discuss solution methodologies in Section 3. We then discuss relevant related works of hierarchical models with our model in Section 4. We give some representative example problems for MMDPs in Section 5 and conclude our paper in Section 6.

## 2 Multi-time Scale MDP

We first present the two time-scale MDP model for simplicity. The $M$ time-scale model with $M > 2$ can be extended from the two time-scale model without difficulty and we will discuss this issue later.

[1]We will discuss this example in more detail in the example problem section.

## 2.1 Model description

The upper level (slow time-scale) MDP has finite state space $I$ and finite action space $\Lambda$. At each (discrete) decision time $n \in \{0, 1, 2, ...,\}$ and at state $i_n \in I$, an action $\lambda_n \in \Lambda$ is taken and $i_n$ makes transition to state $i_{n+1} \in I$ according to probability $P^u(i_{n+1}|i_n, \lambda_n)$. Depending on which action has been taken at which state in the upper level MDP, the lower level (fast time-scale) MDP over one-step slow time-scale period is determined accordingly (what we mean by this will be clearer below). Every MDP in the lower level shares the same state and action space. We denote the finite state space and the finite action space by $X$ and $A$, respectively. We assume that $I \cap X = \emptyset$ and $A \cap \Lambda = \emptyset$, which means that the state and control spaces of the upper level and the lower level are distinct or non-overlapping. We also assume that every control action is admissible at each state in each level for simplicity.

We denote time in the fast time-scale as $t \in \{t_0, t_1, t_2, ...\}$ and $t_{nT} = n$, $n = 0, 1, ...$ and $T$ is a fixed finite scale factor between slow and fast time-scales. We implicitly assume that $t_{nT} = n^+$. That is, there is an infinitesimal gap between $t_{nT}$ and $n$ such that a fast time-scale decision at time $t_{nT}$ is made slightly after a slow time-scale decision at time $n$ has been made.

Let the initial state in the lower level MDP $x \in X$ and the initial state in the upper level MDP $i \in I$ ($x_{t_0} = x$ and $i_0 = i$ at $n = 0$). An action $\lambda_0 \in \Lambda$ will be taken at $i_0$ and the next upper level state $i_1$ will be determined stochastically by $P^u$. Over the time steps of $t_0, t_1, ..., t_{T-1}$, the system follows the lower level MDP evolution. That is, at the state $x$ at $t_0$, an action $a \in A$ is taken and $x$ makes transition to the next state $y \in X$, which is the state at time $t_1$, according to probability $P^l(y|x, a, i, \lambda)$ and the *nonnegative and bounded* reward of $R^l(x, a, i, \lambda)$ is incurred and this process is repeated at the state $y$ at $t_1$, and so forth until the time $t_{T-1}$. That is, the state transition function and the reward function in the lower level MDP (over $T$-epoch) are *induced* by the upper level state and decision. At time $n = 1$, an upper level action $\lambda_1$ will be taken at $i_1$ (this will trigger a new MDP determination) and starting with a state $z$ at $t_T$ (determined stochastically from $P^l(z|x_{t_{T-1}}, a_{t_{T-1}}, i_0, \lambda_0)$ *for now* — we will consider a distribution over $X$ called $\delta$-initialization function later as a method of determining the states of $x_{nT}$ for all $n$), the newly determined lower level MDP evolves (over the next $T$-epoch). See Figure 1 for graphical illustration of time evolution in this process.

Throughout this paper, we will use the term "decision rule" related with infinite horizon and the term "policy" related with finite horizon. Define a lower level *decision rule* $d^l = \{\pi^l_n\}$, $n = 0, 1, ...,$ as a sequence of $T$-horizon *nonstationary policies* defined such that for all $n$, $\pi^l_n = \{\phi_{t_{nT}}, ..., \phi_{t_{(n+1)T-1}}\}$ is a sequence of functions where for all $k \geq 0$, $\phi_{t_k} : X \times I \times \Lambda \rightarrow A$. We will say that a lower level decision rule is *stationary with respect to the slow time-scale* $n$ if $\pi^l_n = \pi^l_{n'}$ for all $n, n'$ and *we will restrict ourselves to only this class of decision rules here.* We will denote the set of all possible such stationary decision rules with respect to the slow time-scale as $\mathcal{D}^l$, and omit the subscript $n$ in $\pi^l_n$

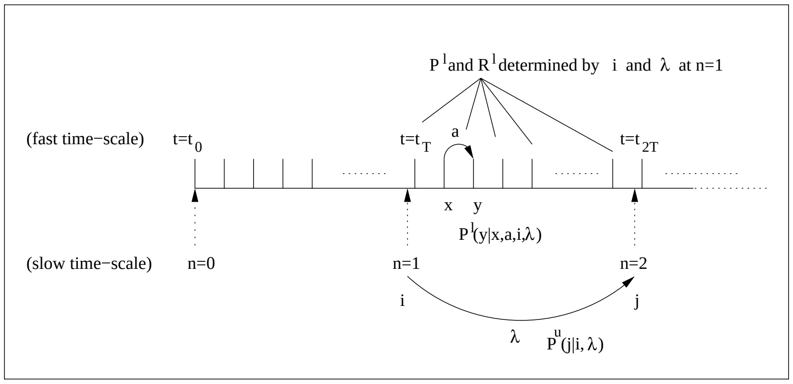

Figure 1: Graphical illustration of time evolution in the two time-scale MDPs

in this case and use the time $t_0, ..., t_{T-1}$ to refer the sequence of functions of $\pi^l$ if necessary, and denote $\Pi^l$ as the set of all possible such $T$-horizon nonstationary policies $\pi^l$. We will also omit the subscript on $\phi$ if $\pi^l$ is stationary (with respect to the fast time-scale).

Given a lower level decision rule $d^l \in \mathcal{D}^l$ and a nonnegative and bounded immediate reward function $\mathcal{I}^u$ defined over $I \times \Lambda$ for the upper level, we define a function $R^u$ such that for all $n \geq 0$, for $x \in X$, $i_n \in I$ and $\lambda_n \in \Lambda$,

$$R^u(x, i_n, \lambda_n, \pi^l) = E^x_{i_n,\lambda_n} \left\{ \sum_{t=t_{nT}}^{t_{(n+1)T-1}} \alpha^{\sigma(t)} R^l(x_t, \phi_t(x_t, i_n, \lambda_n), i_n, \lambda_n) \right\} + \mathcal{I}^u(i_n, \lambda_n), 0 < \alpha \leq 1, \quad (1)$$

where $\sigma(t_{nT+r}) = r$ for all $n$ with $r = 0, 1, ..., T-1$, and the superscript $x$ on $E$ signifies the initial state, $x_{t_{nT}} = x$, and the subscript $i_n, \lambda_n$ on $E$ signifies that $i_n$ and $\lambda_n$ for the expectation are fixed. We will use this notational method throughout the paper. The function $R^u$ is simply the $T$-horizon total expected (discounted) reward of following the $T$-horizon nonstationary policy $\pi^l$ given $i_n \in I$ and $\lambda_n \in \Lambda$ starting with state $x \in X$ with the zero terminal reward function[2] plus an immediate reward of taking an action $\lambda_n$ at the state $i_n$ at the upper level, and is a bounded function.

The total expected (discounted) reward achieved by the lower level $T$-horizon nonstationary policy $\pi^l$ with an immediate reward at the upper level will act as a *single-step reward* for the upper level MDP. Define the upper level *stationary* decision rule $d^u$ as a function $d^u : X \times I \to \Lambda$ and we denote $\mathcal{D}^u$ as the set of all possible such stationary decision rules. Given the initial $x \in X$ and $i \in I$, our goal is to obtain a decision rule pair of $d^l \in \mathcal{D}^l$ and $d^u \in \mathcal{D}^u$ that achieves the following functional value defined over $X \times I$:

$$V^*(x, i) \quad := \quad \max_{d^u \in \mathcal{D}^u} \max_{d^l \in \mathcal{D}^l} E^{x,i} \left\{ \sum_{n=0}^{\infty} \gamma^n R^u(x_{t_{nT}}, i_n, d^u(x_{t_{nT}}, i_n), \pi^l) \right\}, 0 < \gamma < 1.$$

[2]It is our assumption that the initial state for the next epoch in the slow time-scale does not contribute the reward for the previous epoch. However, a terminal reward can be defined by a function over $X$, in which case we need to add the terminal reward term in $R^u$.

$$= \max_{d^u \in \mathcal{D}^u} \max_{d^l \in \mathcal{D}^l} E^{x,i} \left\{ \sum_{n=0}^{\infty} \gamma^n E^{x_{t_{nT}}}_{i_n, \lambda_n} \left[ \sum_{t=t_{nT}}^{t_{(n+1)T-1}} \alpha^{\sigma(t)} R^l \left( x_t, \phi_t(x_t, i_n, d^u(x_{t_{nT}}, i_n)), i_n, d^u(x_{t_{nT}}, i_n) \right) \right] + \mathcal{I}^u(i_n, \lambda_n) \right\} \quad (2)$$

where we will refer to $V^*$ as *the two-level optimal infinite horizon discounted value function.*

The second functional value defined as our objective function is

$$J^*(x,i) := \max_{d^u \in \mathcal{D}^u} \max_{d^l \in \mathcal{D}^l} \lim_{H \to \infty} \frac{1}{H} E^{x,i} \left\{ \sum_{n=0}^{H-1} R^u(x_{t_{nT}}, i_n, d^u(x_{t_{nT}}, i_n), \pi^l) \right\},$$

where we refer to $J^*$ as *the two-level optimal infinite horizon average value function.*

We can see that from the definition of the upper level decision rule, the decisions to be made at the upper level must depend on the lower level state, which is the initial state for the lower level MDP evolution over $T$-horizon in the fast time-scale. The initial state $x_{t_{nT}}, n = 1, 2, ...$ is determined stochastically by following the policy $\pi^l$. We will consider more general case of determining the initial state in the later subsection to expand the flexibility of our model. We also remark that even though we added the immediate reward function $\mathcal{I}^u$ in the definition of $R^u$ to make our model description more natural, the function $R^l$ can "absorb" the function $\mathcal{I}^u$ by newly defining the function $R^l$ itself as

$$\begin{aligned} R^l(x,i,\lambda,\pi^l) &\leftarrow R^l(x,i,\lambda,\pi^l) + \frac{1}{T}\mathcal{I}^u(i,\lambda) \text{ for } \alpha = 1 \text{ and} \\ R^l(x,i,\lambda,\pi^l) &\leftarrow R^l(x,i,\lambda,\pi^l) + \left(\frac{1-\alpha}{1-\alpha^T}\right)\mathcal{I}^u(i,\lambda) \text{ for } 0 < \alpha < 1. \end{aligned}$$

## 2.2 Optimality equations

Because the upper level sequential dynamics is essentially just an MDP with a reward function defined via the lower level MDP dynamics (by fixing a lower level decision rule), with a simple adaptation of the standard MDP theory (see, e.g, [1] [3] [18] or [30]), the following results hold for MMDPs. Therefore, we omit detailed proofs.

The first theorem yields an optimality equation satisfied by $V^*$. We first define a set of all possible $T$-horizon (lower level) nonstationary policies under a fixed pair of an upper level state and an upper level action. For a given pair of $i \in I$ and $\lambda \in \Lambda$,

$$\Pi^l[i,\lambda] := \left\{ \pi^l[i,\lambda] \;\middle|\; \pi^l[i,\lambda] := \{\phi^{i,\lambda}_{t_0}, ..., \phi^{i,\lambda}_{t_{T-1}}\}, \phi^{i,\lambda}_{t_k} : X \times \{i\} \times \{\lambda\} \to A \text{ and } k = 0, ..., T-1 \right\}$$

and let $\mathcal{P}^T_{xy}(\pi^l[i,\lambda])$ the probability that state $y \in X$ is reached by $T$-steps starting with $x$ by following the $T$-horizon nonstationary policy $\pi^l[i,\lambda]$. Note that this probability can be obtained from $P^l$.

**Theorem 2.1** *For all $x \in X$ and $i \in I$,*

$$V^*(x,i) = \max_{\lambda \in \Lambda} \left( \max_{\pi^l[i,\lambda] \in \Pi^l[i,\lambda]} \left\{ R^u(x,i,\lambda,\pi^l[i,\lambda]) + \gamma \sum_{y \in X} \sum_{j \in I} \mathcal{P}^T_{xy}(\pi^l[i,\lambda]) P^u(j|i,\lambda) V^*(y,j) \right\} \right)$$

*and $V^*$ is the unique solution to the above equation. Furthermore, for each pair of $x$ and $i$, let the arguments that achieve the r.h.s of this equation as $\lambda^*$ and $\pi^*[i,\lambda] = \{\phi^*_{t_k}\}$, and set $d^u(x,i) = \lambda^*$ for $d^u$ and set $\pi^l$ such that $\phi_{t_k}(x,i,\lambda^*) = \phi^*_{t_k}(x,i,\lambda^*)$ for $d^l$. The pair of $d^u$ and $d^l$ achieves $V^*$.*

**Proof:** We will define an MDP that operates in the slow time-scale $n$ as follows. The state at time $n$ is a pair of the lower level state and the upper level state, $(x_{t_{nT}}, i_n)$. An action at state $(x_{t_{nT}}, i_n)$ is a composite control of $\lambda_n \in \Lambda$ and $\pi^l[i_n, \lambda_n] \in \Pi^l[i_n, \lambda_n]$ (from our assumption that $t_{nT} = n^+$, $\pi^l[i_n, \lambda_n]$ will be taken slightly after $\lambda_n$ is taken). Observe that we can view $\pi^l$ as one-step action at the slow time-scale. More precisely, the admissible action set for state $(x_{t_{nT}}, i_n)$ is defined as the set given by

$$\{(\lambda, \tau) | \lambda \in \Lambda, \tau \in \Pi^l[i_n, \lambda]\}$$

The transition probability from $(x_{t_{nT}}, i_n)$ to $(x_{t_{(n+1)T}}, i_{n+1})$ is determined directly from $\mathcal{P}^T$ and $P^u$. Then, from the standard MDP theory, for this MDP, we can write Bellman's optimality equation and an optimal decision rule that achieves the unique optimal value at each state is derived, from which we can conclude our result. ∎

Even though we assumed finite state spaces with finite action spaces, the issue of infinite/finite state/action space and bounded/unbounded reward function can be discussed from the well-known MDP theory. We refer [1] for a substantial discussion in this matter. We now state a similar result to the well-known fact for the average reward case in the MDP theory for the function $J^*$ we defined.

**Theorem 2.2** *If there exists a bounded function $\zeta$ defined over $X \times I$ and a constant $g$ such that for all $x \in X$ and $i \in I$,*

$$g + \zeta(x,i) = \max_{\lambda \in \Lambda} \left( \max_{\pi^l[i,\lambda] \in \Pi^l[i,\lambda]} \left\{ R^u(x,i,\lambda,\pi^l[i,\lambda]) + \sum_{y \in X} \sum_{j \in I} \mathcal{P}^T_{xy}(\pi^l[i,\lambda]) P^u(j|i,\lambda) \zeta(y,j) \right\} \right), \tag{3}$$

*then there exists a decision rule pair of $d^u \in \mathcal{D}^u$ and $d^l \in \mathcal{D}^l$ that achieves $J^*(x,i)$ and $g = J^*(x,i)$ for all $x$ and $i$.*

For conditions that make the "if" part of the above theorem hold, refer [1] or [18] for a substantial discussion in the MDP context. An optimal decision rule pair can be obtained by similar way to the method stated in Theorem 2.1.

## 2.3 Initialization function

So far we considered the case where $x_{t_{nT}}, n = 1, 2, ...$ is determined by the $T$-horizon nonstationary policy. In the model we described before, $x_{t_{nT}}, n = 1, 2, ...$ is (stochastically) determined by following the policy $\pi^l$ with given $x_{t_0}$. Considering more general model, we first define an *initialization function* $\delta$. We then determine $x_{t_{nT}}, n = 1, 2, ...$ by the function $\delta$. That is, the lower level MDP initial state for each new $T$-period (in the fast time-scale) is initialized by the function $\delta$. This is motivated by problem specific nature — organizing behavior in a hierarchy.

Here are some examples of $\delta$. As in the previous description of the model, $\delta$ can be a function defined over $X \times I \times \Lambda$ such that for given $x, i, \lambda$, $\delta(x, i, \lambda)$ is a probability distribution over $X$. Given $x \in X$, $i \in I$ and $\lambda \in \Lambda$, we will use the notation of $\delta(x, i, \lambda)[y]$ to denote the probability defined on $y \in X$ by $\delta(x, i, \lambda)$. In the previous model description, $\delta(x, i, \lambda)[y]$ corresponds to $\mathcal{P}^T_{xy}(\pi^l[i, \lambda])$. From now on, we will use the notation $\delta^{\pi^l}$ to explicitly express the dependence on the lower level policy $\pi^l$ if that is the case. Or it can be defined such that the determination of $x_{t_{nT}}$ depends on the state $x_{t_{nT-1}}$. For example, for some $x, y \in X$, $i \in I$, $\lambda \in \Lambda$,

$$\delta^{\pi^l}(x, i, \lambda)[y] = \sum_{z \in X} \mathcal{P}^{T-1}_{xz}(\pi^l[i, \lambda]) \rho(y|z),$$

where $\rho(y|z)$ denotes a probability of choosing $y$ given $z$.

For some cases, the slow time-scale decisions (e.g., “reset” control, etc.) only will affect the new initial lower level state. In this case, $\delta$ is defined over $X \times \Lambda$ such that $\delta(x, \lambda)$ gives a probability distribution over $X$. The very idea of this $\delta$ is parallel to the transition structure in Markovian slowscale model given in [20]. Finally, the determination of $x_{t_{nT}}$ can be independent of $x_{t_{nT}-1}$ or $x_{t_{(n-1)T}}$. For example, we can consider the state in the lower level is initialized depending on the upper level current state $i$ and the next state $j$. $\delta$ is defined over $I \times I$ such that $\delta(i, j)$ for some $i, j \in I$ gives a probability distribution over $X$.

The $\delta$-initialization function gives much more flexibility in our multi-time scale model. Depending on the problem structure, $\delta$ can be defined accordingly. With the introduction of $\delta$, we simply need to rewrite $V^*$ equation (similarly to the $J^*$ case) as

$$V^*(x, i) = \max_{\lambda \in \Lambda} \left( \max_{\pi^l[i,\lambda]} \left\{ R^u(x, i, \lambda, \pi^l[i, \lambda]) + \gamma \sum_{y \in X} \sum_{j \in I} \delta^{\pi^l}(x, i, \lambda)[y] P^u(j|i, \lambda) V^*(y, j) \right\} \right), x \in X, i \in I,$$

where we used the first $\delta$ example function that depends on $\pi^l$. In particular, if the $\delta$-function is independent of $\pi^l[i, \lambda]$ (or we will say that the $\delta$-function is independent of the lower level policies), then we can write the above equation as

$$V^*(x, i) = \max_{\lambda \in \Lambda} \left\{ \max_{\pi^l[i,\lambda]} (R^u(x, i, \lambda, \pi_l)) + \gamma \sum_{y \in X} \sum_{j \in I} \delta(x, i, \lambda)[y] P^u(j|i, \lambda) V^*(y, j) \right\}, x \in X, i \in I.$$

This special case is very interesting because the *optimal* finite $T$-horizon value at the lower level will act as a single step reward for the upper level (along with immediate reward). The upper level decision maker in this case directs/determines a problem at each time that the lower level decision maker needs to solve and the lower level decision maker seeks a "local" optimal solution for $T$-horizon and follows one of the optimal nonstationary policies that achieve the solution. The decision process of how to direct a problem at each time for the upper decision maker will depend on the local optimal performance made by the lower decision maker. In this sense, this has a flavor of the underlying philosophy of the Stackelberg (leader-follower) game (see, e.g, [2]). This is not true in general because $\delta$-initialization function may depend on the lower level policy $\pi^l[i,\lambda]$, where in this case the lower level decision maker needs to choose a policy not only concerned with the local performance of the policy but also effects of the policy in the future performance.

## 2.4 Extension to more than two time-scales

In this subsection, we briefly discuss how to extend the two time-scale model to more than two time-scale model by illustrating the three time-scale model. So we introduce the higher level (referred to as the *highest* level) decision making process that evolves with time $s$ and affects the dynamics of MDP in the $n$ time-scale (denoted as $n \in \{n_0, n_1, ...,\}$) and $n_{sH} = s^+, s = 0, 1, 2, ...$ and $H$ is the finite scale factor. We denote the state and the action space of the highest level as $M$ and $\Psi$, respectively. The notation used in this subsection coincides with the previous sections to avoid confusement.

The transition structures are now defined to be given $x, y \in X$, $i, j \in I$, $m, m' \in M$ and $a \in A$, $\lambda \in \Lambda$, $\psi \in \Psi$, $P^u(j|i, \lambda, m, \psi)$ for the upper level, and $P^l(y|x, a, i, \lambda, m, \psi)$ for the lower level, and $P^h(m'|m, \psi)$ for the highest level. The decision rule for the highest level $d^h$ is defined to be stationary decision rule $d^h : X \times I \times M \to \Psi$, and the the decision rule for the upper level $d^u$ is defined such that $d^u = \{\pi^u_s\}$, $s = 0, 1, ...,$ as a sequence of $H$-horizon *nonstationary policies* where for all $s$, $\pi^u_s = \{\chi_{n_{sH}}, ..., \chi_{n_{(s+1)H-1}}\}$ is a sequence of functions where for all $k \geq 0$, $\chi_{n_k} : X \times I \times M \times \Psi \to \Lambda$. The decision rule $d^u$ is stationary with respect to the time-scale $s$. Similarly, we define the lower level decision rule $d^l = \{\pi^l_n\}$ where $\pi^l_n = \{\phi_{t_{nT}}, ..., \phi_{t_{(n+1)T-1}}\}$ and $\phi_{t_k} : X \times I \times \Lambda \times M \times \Psi \to A$ for all $k \geq 0$. The decision rule $d^l$ is stationary with respect to the time-scale $n$.

The lower level reward function $R^l$ is now a function of $x, a, i, \lambda, m, \psi$ and the upper level single step reward function $R^u$ is now a function of $x, i, \lambda, m, \psi, \pi^l$ and defined from $R^l$ with an immediate reward function at the upper level. That is, the highest level state and decision will affect the upper and lower MDP reward functions. The single step reward function $R^h$ for the highest level is a function of $x, i, m, \pi^u, \pi^l$ and defined from $R^u$ over $H$-horizon in the time-scale $n$ and an immediate reward function at the highest level. From this reward functions, we can define the three level optimal value function and determine the three level optimality equation.

## 3 Solving MMDPs

The methods of obtaining the optimal decision rule for each level in MMDPs are well-established via the well-known MDP theory. We will pay attention to $\delta$-initialization function that depends on the lower level policies and is defined over $X \times I \times \Lambda$ such that $\delta^{\pi^l}(x,i,\lambda)$ for $x \in X$, $i \in I$, and $\lambda \in \Lambda$ gives a probability distribution over $X$. The discussion here can be easily extended to other $\delta$-functions.

### 3.1 Exact methods

We first discuss discounted case and then average case. Define an operator $\Theta$ such that for a (bounded and measurable) function $V$ defined over $X \times I$,

$$\Theta(V)(x,i) = \max_{\lambda \in \Lambda} \left( \max_{\pi^l[i,\lambda] \in \Pi^l[i,\lambda]} \left\{ R^u(x,i,\lambda,\pi^l[i,\lambda]) + \gamma \sum_{y \in X} \sum_{j \in I} \delta^{\pi^l}(x,i,\lambda)[y] P^u(j|i,\lambda) V(y,j) \right\} \right) \tag{4}$$

for all $x$ and $i$. Then, $\Theta$ is a $\gamma$-*contraction-mapping* in sup-norm. For any function $V$ defined over $X \times I$, let $\|V\| = \sup_{x,i} |V(x,i)|$. For any bounded and measurable two function $U$ and $V$ defined over $X \times I$, it is true that

$$\|\Theta(U) - \Theta(V)\| \leq \gamma \|U - V\|.$$

This implies that $V^*(x,i)$ is unique from the well-known fixed point theorem. Furthermore, for any such $V$,

$$\Theta^n(V) \to V^* \text{ as } n \to \infty,$$

where this method is known as *value iteration.*

For the average reward case, we assume that (appropriately modified) one of the ergodicity conditions in page 56 [18] holds. Then, average reward value iteration can be also applied [18]. Let $\Phi$ be an operator that maps a function $V$ defined over $X \times I$ to another function defined over $X \times I$ given by

$$\Phi(V)(x,i) = \max_{\lambda \in \Lambda} \left( \max_{\pi^l[i,\lambda] \in \Pi^l[i,\lambda]} \left\{ R^u(x,i,\lambda,\pi^l[i,\lambda]) + \sum_{y \in X} \sum_{j \in I} \delta^{\pi^l}(x,i,\lambda)[y] P^u(j|i,\lambda) V(y,j) \right\} \right) \tag{5}$$

for all $x$ and $i$. Then, with an arbitrary (bounded and measurable) function $V$ defined over $X \times I$, for all $x \in X, i \in I$,

$$\Phi^n(V)(x,i) - \Phi^{n-1}(V)(x,i) \to g \text{ as } n \to \infty$$

and for any fixed state pair $y \in X$ and $j \in I$,

$$\Phi^n(V)(x,i) - \Phi^n(V)(y,j) \to \zeta(x,i) \text{ as } n \to \infty, x \in X, i \in I.$$

We can also use *"policy iteration"* once $R^u$ is determined. See, e.g., [26] for average reward case and [30] for discounted case.

## 3.2 Approximation methods

There are numerous approximation algorithms to solve MDPs. We resort to the books by Puterman [30] or by Bertsekas and Tsitsiklis [5], etc. for a substantial discussion. In this section, we analyze the performance of an approximation-based scheme for solving MMDPs.

Our first approximation is on the $\delta$-initialization function. One of the main difficulties to obtain an optimal decision rule pair would be the given initialization function's possible dependence on the lower level nonstationary policies. Suppose that is true and consider a $\delta'$-initialization function that is independent of the lower level policies and approximates the given $\delta^{\pi^l}$-initialization function with respect to a given metric. Then there exists a unique function $\hat{V}^*$ defined over $X \times I$ such that for all $x$ and $i$,

$$\hat{V}^*(x,i) = \max_{\lambda \in \Lambda} \left\{ \max_{\pi^l[i,\lambda] \in \Pi^l[i,\lambda]} \left( R^u(x,i,\lambda,\pi^l[i,\lambda]) \right) + \gamma \sum_{y \in X} \sum_{j \in I} \delta'(x,i,\lambda)[y] P^u(j|i,\lambda) \hat{V}^*(y,j) \right\}.$$

We can bound then $|\hat{V}^*(x,i) - V^*(x,i)|$ for all $x$ and $i$ by Theorem 4.2 in the Müller's work [27] with a metric called "integral probability metric" on the difference between $\delta'$ and $\delta^{\pi^l}$. Of course, if the MMDP problem to solve is associated with the lower level policy independent $\delta$-function, we wouldn't need this approximation step.

The second approximation is on the value of $R^*$ defined as

$$R^*(x,i,\lambda) = \max_{\pi^l[i,\lambda] \in \Pi^l[i,\lambda]} \left( R^u(x,i,\lambda,\pi^l[i,\lambda]) \right)$$

and on $\hat{V}^*(x,i)$. It will be often impossible to get the true $R^*$ due to the huge state space size of the lower level and even more computationally cumbersome if $T$ is relatively large even if $|X|$ is a moderate sized number even though theoretically we can use "backward induction". Obtaining the true value of the function $\hat{V}^*$ is also almost infeasible in many cases with similar reasons. Suppose we approximate $R^*$ by $\hat{R}$ such that

$$\sup_{x,i,\lambda} |R^*(x,i,\lambda) - \hat{R}(x,i,\lambda)| \leq \kappa$$

and $\hat{V}^*$ by some bounded and measurable function $U$ defined over $X \times I$ such that

$$\sup_{x,i} |\hat{V}^*(x,i) - U(x,i)| \leq \epsilon.$$

We will discuss an example of such $\hat{R}$ and of the function $U$ later in this subsection. Now define a stationary (upper level) decision rule $d^u$ such that for all $x \in X$ and $i \in I$,

$$d^u(x,i) = \arg\max_{\lambda \in \Lambda} \left( \hat{R}(x,i,\lambda) + \gamma \sum_{y \in X} \sum_{j \in I} \delta'(x,i,\lambda)[y] P^u(j|i,\lambda) U(y,j) \right).$$

Our goal is to bound the performance of the decision rule $d^u$ from $\hat{V}^*$. We define the value of following the decision rule $d^u$ given an initialization function $\delta'$ as follows:

$$\hat{V}(x,i) = E^{x,i}_{\delta'} \left\{ \sum_{n=0}^{\infty} \gamma^n \hat{R}(x_{t_{nT}}, i_n, d^u(x_{t_{nT}}, i_n)) \right\},$$

where we used (by abusing the notation) $E_{\delta'}$ to indicate that $x_{t_{nT}}, n = 1, 2, ...$ is a random variable denoting (lower level) state at time $t_{nT}$ determined stochastically by $\delta'$. We now state a performance bound as a theorem below.

**Theorem 3.1** *If* $\sup_{x,i,\lambda} |R^*(x,i,\lambda) - \hat{R}(x,i,\lambda)| \le \kappa$ *and* $\sup_{x,i} |\hat{V}^*(x,i) - U(x,i)| \le \epsilon$,

$$|\hat{V}^*(x,i) - \hat{V}(x,i)| \le \frac{2\gamma\epsilon + \kappa}{1-\gamma} \text{ for all } x \in X \text{ and } i \in I.$$

**Proof:** Let the argument that achieves the r.h.s of Equation (4) with replacing $\delta^{\pi^l}$ by $\delta'$ be $\lambda_U$ for a function $U$. We will use the notation $\Theta'$ for this replacement. From the contraction mapping property of the $\Theta'$ operator, for all $x \in X$ and $i \in I$,

$$|\Theta'(\hat{V}^*)(x,i) - \Theta'(U)(x,i)| \le \gamma \cdot \sup_{x,i} |\hat{V}^*(x,i) - U(x,i)| \le \gamma\epsilon.$$

We show that $|\Theta'(U)(x,i) - \hat{V}(x,i)| \le \frac{\gamma\epsilon(1+\gamma)+\kappa}{1-\gamma}$ for all $x \in X$ and $i \in I$. It then follows that from $\Theta'(\hat{V}^*) = \hat{V}^*$,

$$\begin{aligned} |\hat{V}^*(x,i) - \hat{V}(x,i)| &\le |\Theta'(\hat{V}^*)(x,i) - \Theta'(U)(x,i)| + |\Theta'(U)(x,i) - \hat{V}(x,i)| \\ &\le \gamma\epsilon + \frac{\gamma\epsilon(1+\gamma)+\kappa}{1-\gamma} = \frac{2\gamma\epsilon+\kappa}{1-\gamma}, \end{aligned}$$

which gives the desired result.

Now, for all $x \in X$ and $i \in I$,

$$\begin{aligned} \Theta'(U)(x,i) &= R^*(x,i,\lambda_U) + \gamma \sum_{y \in X} \sum_{j \in I} \delta'(x,i,\lambda_U)[y] P^u(j|i,\lambda_U) U(y,j) \text{ by the definition of } \Theta' \\ &\le \hat{R}(x,i,\lambda_U) + \kappa + \gamma \sum_{y \in X} \sum_{j \in I} \delta'(x,i,\lambda_U)[y] P^u(j|i,\lambda_U) U(y,j) \text{ by the given assumption} \\ &\le \hat{R}(x,i,d^u(x,i)) + \kappa + \gamma \sum_{y \in X} \sum_{j \in I} \delta'(x,i,d^u(x,i))[y] P^u(j|i,d^u(x,i)) U(y,j) \\ &\qquad \text{by the definition of } d^u \end{aligned}$$

$$\begin{aligned}
&\leq \hat{R}(x,i,d^u(x,i)) + \kappa + \gamma \sum_{y\in X}\sum_{j\in I} \delta'(x,i,d^u(x,i))[y]P^u(j|i,d^u(x,i))[\hat{V}^*(y,j)+\epsilon] \\
&= \hat{R}(x,i,d^u(x,i)) + \gamma \sum_{y\in X}\sum_{j\in I} \delta'(x,i,d^u(x,i))[y]P^u(j|i,d^u(x,i))\hat{V}^*(y,j) + \gamma\epsilon + \kappa \\
&\leq \hat{R}(x,i,d^u(x,i)) + \gamma \sum_{y\in X}\sum_{j\in I} \delta'(x,i,d^u(x,i))[y]P^u(j|i,d^u(x,i))[\Theta'(U)(y,j)+\gamma\epsilon] + \gamma\epsilon + \kappa \\
&= \hat{R}(x,i,d^u(x,i)) + \gamma \sum_{y\in X}\sum_{j\in I} \delta'(x,i,d^u(x,i))[y]P^u(j|i,d^u(x,i))\Theta'(U)(y,j) + \gamma\epsilon(1+\gamma) + \kappa \\
&\leq \hat{R}(x,i,d^u(x,i)) + \gamma \sum_{y\in X}\sum_{j\in I} \delta'(x,i,d^u(x,i))[y]P^u(j|i,d^u(x,i))\Big[\hat{R}(y,j,d^u(y,j)) + \kappa \\
&\qquad + \gamma \sum_{z\in X}\sum_{k\in I} \delta'(y,j,d^u(y,j))[z]P^u(k|j,d^u(y,j))U(z,k)\Big] + \gamma\epsilon(1+\gamma) + \kappa \\
&= \hat{R}(x,i,d^u(x,i)) + \gamma \sum_{y\in X}\sum_{j\in I} \delta'(x,i,d^u(x,i))[y]P^u(j|i,d^u(x,i))\hat{R}(y,j,d^u(y,j)) \\
&\qquad + \gamma^2 \sum_{y\in X}\sum_{j\in I} \delta'(x,i,d^u(x,i))[y]P^u(j|i,d^u(x,i)) \\
&\qquad \times\Big[\sum_{z\in X}\sum_{k\in I} \delta'(y,j,d^u(y,j))[z]P^u(k|j,d^u(y,j))U(z,k)\Big] + \gamma\epsilon(1+\gamma) + \kappa(1+\gamma) \\
&\leq \hat{R}(x,i,d^u(x,i)) + \gamma \sum_{y\in X}\sum_{j\in I} \delta'(x,i,d^u(x,i))[y]P^u(j|i,d^u(x,i))\hat{R}(y,j,d^u(y,j)) \\
&\qquad + \gamma^2 \sum_{y\in X}\sum_{j\in I} \delta'(x,i,d^u(x,i))[y]P^u(j|i,d^u(x,i)) \\
&\qquad \times\Big[\sum_{z\in X}\sum_{k\in I} \delta'(y,j,d^u(y,j))[z]P^u(k|j,d^u(y,j))\Theta'(U)(z,k)\Big] \\
&\qquad + \gamma^2\epsilon(1+\gamma) + \gamma\epsilon(1+\gamma) + \kappa(1+\gamma)
\end{aligned}$$

Keep iterating (under the sum sign) this way, we have that for all $l = 0, 1, ...,$ and $x \in X, i \in I$,

$$\begin{aligned}
\Theta'(U)(x,i) &\leq E^{x,i}_{\delta'}\left[\sum_{n=0}^{l} \gamma^n \hat{R}(x_{t_{nT}}, i_n, d^u(x_{t_{nT}}, i_n))\right] + \gamma^{l+1} E_{\delta'}[\Theta'(U)(x_{t_{(l+1)T}}, i_{l+1})] \\
&\qquad + \gamma\epsilon(1+\gamma) + \cdots + \gamma^{l+1}\epsilon(1+\gamma) + \kappa(1+\gamma+\cdots+\gamma^l). \qquad (6)
\end{aligned}$$

Since $\Theta'(U)$ is bounded, the second term on the r.h.s. of Equation (6) converges to zero as $l \to \infty$ and the first term becomes $\hat{V}(x,i)$. Therefore it follows that $\Theta'(U)(x,i) - \hat{V}(x,i) \leq \frac{\gamma\epsilon(1+\gamma)+\kappa}{1-\gamma}$. This proves the upper bound case.

For the lower bound case, observe that for all $x \in X$ and $i \in I$,

$$\begin{aligned}
\Theta'(U)(x,i) &= R^*(x,i,\lambda_U) + \gamma \sum_{y\in X}\sum_{j\in I} \delta'(x,i,\lambda_U)[y]P^u(j|i,\lambda_U)U(y,j) \\
&\geq R^*(x,i,d^u(x,i)) + \gamma \sum_{y\in X}\sum_{j\in I} \delta'(x,i,d^u(x,i))[y]P^u(j|i,d^u(x,i))U(y,j) \\
&\qquad \text{by definition of } \lambda_U
\end{aligned}$$

$$\geq \hat{R}(x,i,d^u(x,i)) - \kappa + \gamma \sum_{y\in X}\sum_{j\in I} \delta'(x,i,d^u(x,i))[y]P^u(j|i,d^u(x,i))[\hat{V}^*(y,j)-\epsilon].$$

By the similar arguments for the upper bound case, we can then show that $\Theta'(U)(x,i) - \hat{V}(x,i) \geq -\frac{\gamma\epsilon(1+\gamma)+\kappa}{1-\gamma}$. This concludes our proof. ∎

We remark that a related work for this theorem can be found in Corollary 1 in [35] with the assumption of the finite state space and the result of the work only gives an upper bound. Our analysis takes a totally different approach and can be applied to *Borel* state space even though our proof shows for the countable case. Furthermore, the result gives not only a lower bound but also a tighter bound. Now we give an example of $\hat{R}$. *From now on, we assume that the lower level reward function $R^l$ is defined such that it absorbs the upper level immediate reward function $\mathcal{I}^u$* as we discussed in subsection 2.1. Our approximation uses a heuristic lower level policy $\pi^l$ that guarantees the $T$-horizon total expected discounted reward of following the policy $\pi^l$ is within an error bound from the optimal finite-horizon value.

The methodology of the example is the *rolling horizon* approach [19] where we choose a horizon $h \ll T$ and solve for the optimal $h$-horizon total expected discounted reward and we define a (greedy) stationary policy with respect to the value function. We begin by defining $h$-horizon total expected discounted reward with $h = 1, ..., T$ for every given $i \in I$ and $\lambda \in \Lambda$:

$$R^*_h(x,i,\lambda) = \max_{\pi^l[i,\lambda]} E^x_{i,\lambda}\left\{\sum_{t=0}^{h-1}\alpha^t R^l(x_t, \phi_t^{i,\lambda}(x_t,i,\lambda),i,\lambda)\right\}, 0<\alpha<1, i\in I, \lambda\in\Lambda, \tag{7}$$

where $R^*_0(x,i,\lambda) = 0$ for all $x \in X$. We also let $R^{\pi^l}(x,i,\lambda) = R^u(x,i,\lambda,\pi^l[i,\lambda])$ defined in Equation (1) for every $i$ and $\lambda$ with $0<\alpha<1$ and $R_{\max} = \max_{x,a,i,\lambda} R^l(x,a,i,\lambda)$.

**Proposition 3.1** *For every given $i \in I$ and $\lambda \in \Lambda$ and a selected $h$ in $\{1, ..., T\}$, define a lower level stationary policy $\pi^l[i,\lambda]$ as*

$$\phi^{i,\lambda}(x,i,\lambda) = \arg\max_{a\in A}\left(R^l(x,a,i,\lambda) + \alpha\sum_{y\in X}P^l(y|x,a,i,\lambda)R^*_{h-1}(y,i,\lambda)\right) \text{ for all } x\in X.$$

*Then, for all $x, i, \lambda$,*

$$0 \leq R^*(x,i,\lambda) - R^{\pi^l}(x,i,\lambda) \leq \frac{R_{\max}\alpha^h(1-\alpha^T)}{1-\alpha}.$$

**Proof:** The lower bound is from the definition of $R^*$. Fix arbitrary $i \in I$ and $\lambda \in \Lambda$. Define an operator $\Omega$ that maps a (bounded) function $V$ defined over $X$ to another function defined over $X$ given by

$$\Omega(V)(x) = \max_{a\in A}\left(R^l(x,a,i,\lambda) + \alpha\sum_{y\in X}P^l(y|x,a,i,\lambda)V(y)\right) \tag{8}$$

It is well-known that $R^*_h = \Omega^h(R^*_0)$ (see, e.g., [3] [18], etc.). By the contraction mapping property of $\Omega$, (with $\|f\| = \sup_{x,i,\lambda} |f(x,i,\lambda)|$),

$$\begin{aligned} \|R^*_T - R^*_h\| &\leq \alpha \|R^*_{T-1} - R^*_{h-1}\| \leq \cdots \leq \alpha^h \|R^*_{T-h} - R^*_0\| \\ &\leq \alpha^h (1 + \alpha + \cdots + \alpha^{T-h-1}) R_{\max} \leq \frac{R_{\max}(\alpha^h - \alpha^T)}{1-\alpha}. \end{aligned} \quad (9)$$

Now by the definition of $\pi^l[i,\lambda]$ (the following proof idea is from [19]),

$$\begin{aligned} R^*_h(x,i,\lambda) &= R^l(x, \phi^{i,\lambda}(x,i,\lambda), i, \lambda) + \alpha \sum_y P^l(y|x, \phi^{i,\lambda}(x,i,\lambda), i, \lambda) R^*_{h-1}(y,i,\lambda) \\ &\leq R^l(x, \phi^{i,\lambda}(x,i,\lambda), i, \lambda) + \alpha \sum_y P^l(y|x, \phi^{i,\lambda}(x,i,\lambda), i, \lambda) R^*_h(y,i,\lambda) \end{aligned}$$

Keep iterating under the sum sign, we have that for all $w = 0, 1, ..., T-1$ and for all $x \in X$,

$$R^*_h(x,i,\lambda) \leq E^x_{i,\lambda}\left[\sum_{t=0}^{w} \alpha^t R^l(x_t, \phi^{i,\lambda}(x_t,i,\lambda), i, \lambda)\right] + \alpha^{w+1} E_{i,\lambda}[R^*_h(x_{w+1}, i, \lambda)]. \quad (10)$$

We let $w = T-1$. It follows then that from the previous inequality (10),

$$R^*_h(x,i,\lambda) \leq R^{\pi^l}(x,i,\lambda) + \frac{R_{\max}\alpha^T(1-\alpha^h)}{1-\alpha}.$$

Therefore, we have

$$R^*(x,i,\lambda) - R^{\pi^l}(x,i,\lambda) \leq R^*(x,i,\lambda) - R^*_h(x,i,\lambda) + \frac{R_{\max}\alpha^T(1-\alpha^h)}{1-\alpha}.$$

Combining the result in Equation (9) with the previous inequality, we finally have that

$$R^*(x,i,\lambda) - R^{\pi^l}(x,i,\lambda) \leq \frac{R_{\max}\alpha^h(1-\alpha^T)}{1-\alpha}.$$

∎

For every given $\kappa > 0$, letting $\kappa \geq \frac{R_{\max}\alpha^h(1-\alpha^T)}{1-\alpha}$ gives the rolling horizon size for a desired error bound for $R^*$. We remark that by letting $T \to \infty$, the above result precisely gives the result of Theorem 3.1 in [19]. The similar approach can be taken for the upper level MDP. We can choose a fixed horizon and use the horizon as the rolling horizon. The value function defined by the horizon approximates $\hat{V}^*$, i.e., an example of $U$. If both levels use the rolling horizon approach, we have two-level approximation. We can easily draw an error bound of this two-level rolling horizon approach from the results obtained in this subsection. In practice, getting the true value of $R^*_h$ will be also difficult even though $h$ is small due to the curse of dimensionality. A way of getting away with a large state space is to use a sampling method to approximate $R^*_h$ probabilistically. See, e.g., [21] in this direction and an analysis for discounted case.

For the average reward case, we consider the case where one of the ergodicity conditions in page 56 [18] holds. Furthermore, we assume that the similar approximation to the first approximation for the discounted case is done by a $\delta'$-initialization function that is independent of the lower level policies and approximates the given $\delta^{\pi^l}$-initialization function. That is, there exists a constant $\hat{g}$ and a function $\hat{\zeta}$ such that for all $x$ and $i$,

$$\hat{g} + \hat{\zeta}(x,i) = \max_{\lambda \in \Lambda} \left\{ \max_{\pi^l[i,\lambda] \in \Pi^l[i,\lambda]} \Big( R^u(x,i,\lambda,\pi^l[i,\lambda]) \Big) + \sum_{y \in X} \sum_{j \in I} \delta'(x,i,\lambda)[y] P^u(j|i,\lambda) \hat{\zeta}(y,j) \right\}$$

and that $|\hat{g} - g|$ is bounded with respect to the degree of the approximation by $\delta'$ for $\delta^{\pi^l}$.

We focus on the second approximation for the average case. We will denote $R_h^*$ defined in Equation (7) with $\alpha = 1$ as $\bar{R}_h^*$ and $\bar{R}^{\pi^l} = R^u(x,i,\lambda,\pi^l[i,\lambda])$ defined in Equation (1) with $\alpha = 1$, and the operator $\Omega$ in Equation (8) with $\alpha = 1$ as $\bar{\Omega}$. Suppose that we approximate $\bar{R}^* (= \bar{R}_T^*)$ by $\hat{R}$ as before such that

$$\sup_{x,i,\lambda} |\bar{R}^*(x,i,\lambda) - \hat{R}(x,i,\lambda)| \leq \kappa$$

and that $\hat{\zeta}$ is approximated by some bounded and measurable function $U$ defined over $X \times I$ such that

$$\sup_{x,i} |\hat{\zeta}(x,i) - U(x,i)| \leq \epsilon.$$

Define a stationary (upper level) decision rule $d^u$ such that for all $x \in X$ and $i \in I$,

$$d^u(x,i) = \arg\max_{\lambda \in \Lambda} \left( \hat{R}(x,i,\lambda) + \sum_{y \in X} \sum_{j \in I} \delta'(x,i,\lambda)[y] P^u(j|i,\lambda) U(y,j) \right).$$

The value of following the decision rule $d^u$ given an initialization function $\delta'$ is defined as follows:

$$\hat{J}(x,i) = \lim_{H \to \infty} \frac{1}{H} E_{\delta'}^{x,i} \left\{ \sum_{n=0}^{H-1} \hat{R}(x_{t_{nT}}, i_n, d^u(x_{t_{nT}}, i_n)) \right\}.$$

We now state a performance bound as a theorem below.

**Theorem 3.2** *Assume that one of the ergodicity conditions in page 56 in [18] holds. If* $\sup_{x,i,\lambda} |\bar{R}^*(x,i,\lambda) - \hat{R}(x,i,\lambda)| \leq \kappa$ *and* $\sup_{x,i} |\hat{\zeta}(x,i) - U(x,i)| \leq \epsilon$,

$$|\hat{g} - \hat{J}(x,i)| \leq 2\epsilon + \kappa \text{ for all } x \in X \text{ and } i \in I.$$

The proof of this theorem can be done via adaptation of the proof of Theorem 3.2 in [10] (using the invariant probability distribution) so we omit the proof. We now provide a counterpart result to Proposition 3.1 for the undiscounted case ($\alpha = 1$) under an ergodicity assumption.

Define $C := \{(x,a) | x \in X, a \in A\}$. For every given $i \in I$ and $\lambda \in \Lambda$, we define $R^l(c,i,\lambda) := R^l(x,a,i,\lambda)$ and $P^l(y|c,i,\lambda) := P^l(y|x,a,i,\lambda)$ for all $c \in C$.

**Assumption 3.1** *There exists a positive number $\nu < 1$ such that for every given $i$ and $\lambda$,*

$$\sup_{c,c' \in C} \sum_{y \in X} |P^l(y|c,i,\lambda) - P^l(y|c',i,\lambda)| \leq 2\nu,$$

We give a performance bound of the rolling horizon policy in terms of span semi-norm; for a bounded function $V$ defined over $X \times I \times \Lambda$ and fixed $i \in I$ and $\lambda \in \Lambda$ (with abusement of notation), $\text{sp}(V) = \sup_x V(x,i,\lambda) - \inf_x V(x,i,\lambda)$.

**Proposition 3.2** *Assume that the ergodicity condition 3.1 holds. For every given $i \in I$ and $\lambda \in \Lambda$ and a selected $h$ in $\{1,...,T\}$, define a lower level stationary policy $\pi^l$ as*

$$\phi^{i,\lambda}(x,i,\lambda) = \arg\max_{a \in A} \left( R^l(x,a,i,\lambda) + \sum_{y \in X} P^l(y|x,a,i,\lambda)\bar{R}^*_{h-1}(y,i,\lambda) \right) \text{ for all } x \in X.$$

*Then, for all $i$ and $\lambda$,*

$$\text{sp}(\bar{R}^* - \bar{R}^{\pi^l}) \leq T \cdot \frac{2\nu^{h-1}R_{\max}}{1-\nu} + \frac{2(\nu^h - \nu^T)R_{\max}}{(1-\nu)^2}$$

**Proof:** We begin with a slightly modified version of Theorem 4.8(a) [18] by Lemma below. See the proof there.

**Lemma 3.1** *Assume that the ergodicity condition 3.1 holds. For every given $i \in I$ and $\lambda \in \Lambda$ and $h = 1,...,T$, there exists a constant $j^*$ such that for all $x \in X$,*

$$\begin{aligned} &(a) \quad \bar{R}^*_h(x,i,\lambda) - \bar{R}^*_{h-1}(x,i,\lambda) \geq \tfrac{-\nu^{h-1}R_{\max}}{1-\nu} + j^* \\ &(b) \quad \bar{R}^*_h(x,i,\lambda) - \bar{R}^*_{h-1}(x,i,\lambda) \leq \tfrac{\nu^{h-1}R_{\max}}{1-\nu} + j^* \end{aligned}$$

Fix $i$ and $\lambda$. Let $\rho_1 = \frac{-\nu^{h-1}R_{\max}}{1-\nu} + j^*$ and $\rho_2 = \frac{\nu^{h-1}R_{\max}}{1-\nu} + j^*$. With a similar reasoning in the proof of Proposition 3.1 and with the inequality in Lemma 3.1(a), we can deduce that for all $w = 0,1,...,T-1$ and for all $x \in X$,

$$\bar{R}^*_h(x,i,\lambda) \leq E^x_{i,\lambda}\left[\sum_{t=0}^{w} R^l(x_t, \phi^{i,\lambda}(x_t,i,\lambda),i,\lambda)\right] + E_{i,\lambda}[\bar{R}^*_h(x_{w+1},i,\lambda)] - (w+1)\rho_1.$$

We let $w = T-1$. It follows then that from the previous inequality,

$$\bar{R}^*_h(x,i,\lambda) \leq \bar{R}^{\pi^l}(x,i,\lambda) + E_{i,\lambda}[\bar{R}^*_h(x_T,i,\lambda)] - T\rho_1.$$

By the same arguments, we have that

$$\bar{R}^*_h(x,i,\lambda) \geq \bar{R}^{\pi^l}(x,i,\lambda) + E_{i,\lambda}[\bar{R}^*_h(x_T,i,\lambda)] - T\rho_2.$$

Combining the above two inequalities, it follows that

$$\text{sp}(\bar{R}^*_h - R^{\pi^l}) \leq T(\rho_2 - \rho_1) = T \cdot \frac{2\nu^{h-1}R_{\max}}{1-\nu}. \tag{11}$$

Now, from the span semi-norm contraction property of $\bar{\Omega}$ [18], we have that

$$\mathrm{sp}(\bar{R}^*_T - \bar{R}^*_h) \le \nu \, \mathrm{sp}(\bar{R}^*_{T-1} - \bar{R}^*_{h-1}) \le \cdots \le \nu^h \, \mathrm{sp}(\bar{R}^*_{T-h}). \tag{12}$$

From Lemma 3.1, we can also deduce that for all $x \in X$,

$$-\frac{R_{\max}(1-\nu^h)}{(1-\nu)^2} + hj^* \le \bar{R}^*_h(x,i,\lambda) \le \frac{R_{\max}(1-\nu^h)}{(1-\nu)^2} + hj^*.$$

Therefore, $\mathrm{sp}(\bar{R}^*_{T-h}) \le \frac{2R_{\max}(1-\nu^{T-h})}{(1-\nu)^2}$. Combining Equation (11) and (12) with the previous inequality, we have the desired result:

$$\mathrm{sp}(\bar{R}^* - \bar{R}^{\pi^l}) \le T \cdot \frac{2\nu^{h-1}R_{\max}}{1-\nu} + \frac{2(\nu^h - \nu^T)R_{\max}}{(1-\nu)^2}. \tag{13}$$

∎

We remark that the above result also gives a bound on finite horizon *average* reward by dividing the both hand sides of Equation (13) by the horizon $T$. In particular, the result by letting $T \to \infty$ in this case does not coincide exactly with the result obtained in Theorem 5.1 in [19] — our result is loose by a factor of 2 in terms of span semi-norm even though the upper bound part in Theorem 5.1 would be the same. This is because the lower bound on the result of Theorem 5.1 is 0 incorporating the fact that the infinite horizon average reward of any stationary decision rule is no bigger than the optimal infinite horizon average reward, where we couldn't take advantage of the fact in our proof steps.

Suppose we have a lower level policy dependent initialization function and we now know that the set of local optimal lower level policies that solve the lower level MDP problem for given $i \in I$ and $\lambda \in \Lambda$. As we can observe, a lower level decision rule determined from these policies doesn't necessarily achieve the optimal multi-level value because it is a locally optimal or greedy choice. However, solving the optimality equation given in Theorem 2.1, for example, is difficult because the size of the set $\Pi^l[i,\lambda]$ is often huge. We should somehow utilize the fact that we know the local optimal lower level policies. To illustrate this, we study this case for discounted case only here. For this purpose, let $\Pi^*[i,\lambda]$ set of $\pi^l[i,\lambda]$'s that solve the lower level MDP problem for given $i \in I$ and $\lambda \in \Lambda$, i.e., achieving $R^*$. We then define a pair of upper and lower level decision rules, $\tilde{d}^u$ and $\tilde{d}^l$, from the arguments that achieve the following equation:

$$\max_{\lambda\in\Lambda}\left(\max_{\pi^l[i,\lambda]\in\Pi^*[i,\lambda]}\left\{R^u(x,i,\lambda,\pi^l[i,\lambda]) + \gamma\sum_{y\in X}\sum_{j\in I}\delta^{\pi^l}(x,i,\lambda)[y]P^u(j|i,\lambda)V^*(y,j)\right\}\right)$$

such that we set $\tilde{d}^u(x,i) = \tilde{\lambda}$ and set $d^l = \{\tilde{\pi}^l\}$, where $\tilde{\lambda}$ and $\tilde{\pi}^l[i,\tilde{\lambda}]$ are the arguments that achieve the above equation. We let the two-level value of following the pair of $\tilde{d}^u$ and $\tilde{d}^l$ be $\tilde{V}(x,i)$. We also let the optimal pair of upper and lower level policies that achieve $V^*(x,i)$ for all $x$ and $i$ as $d^u_*$ and $d^l_*$ (associated with $\pi^l_*$). Our goal is to bound the error between these two value functions.

Let us first impose an ergodicity assumption that there exists a positive number $\mu < 1$ such that for any $x, x'$ and $i, i'$ and for any $\lambda, \lambda'$ and any $\pi, \pi' \in \Pi^l$,

$$\sum_{y \in X} \sum_{j \in I} \left| \delta^{\pi}(x,i,\lambda)[y] P^u(j|i,\lambda) - \delta^{\pi'}(x',i',\lambda')[y] P^u(j|i',\lambda') \right| \leq 2\mu.$$

Observe that for every $x$ and $i$,

$$\begin{aligned}
&R^u(x,i,d^u_*(x,i),\pi^l_*[i,d^u_*(x,i)]) + \gamma \sum_{y \in X} \sum_{j \in I} \delta^{\pi^l_*}(x,i,d^u_*(x,i))[y] P^u(j|i,d^u_*(x,i)) V^*(y,j) \\
&\quad \leq R^u(x,i,d^u_*(x,i),\tilde{\pi}^l[i,d^u_*(x,i)]) + \gamma \sum_{y \in X} \sum_{j \in I} \delta^{\pi^l_*}(x,i,d^u_*(x,i))[y] P^u(j|i,d^u_*(x,i)) V^*(y,j) \\
&\qquad \text{by the definition of } \tilde{\pi}^l[i,d^u_*(x,i)] \\
&\quad \leq R^u(x,i,d^u_*(x,i),\tilde{\pi}^l[i,d^u_*(x,i)]) + \gamma \sum_{y \in X} \sum_{j \in I} \delta^{\tilde{\pi}^l}(x,i,d^u_*(x,i))[y] P^u(j|i,d^u_*(x,i)) V^*(y,j) + \gamma\mu \cdot \mathrm{sp}(V^*) \\
&\qquad \text{by incorporating the ergodicity condition (see, e.g., page 60 in [18])} \\
&\quad \leq R^u(x,i,\tilde{d}^u(x,i),\tilde{\pi}^l[i,\tilde{d}^u(x,i)]) + \gamma \sum_{y \in X} \sum_{j \in I} \delta^{\tilde{\pi}^l}(x,i,\tilde{d}^u(x,i))[y] P^u(j|i,\tilde{d}^u(x,i)) V^*(y,j) + \gamma\mu \cdot \mathrm{sp}(V^*) \\
&\qquad \text{by the definitions of } \tilde{d}^u(x,i) \text{ and } \tilde{\pi}^l[i,\tilde{d}^u(x,i)]
\end{aligned}$$

Then, it immediately follows that for all $x$ and $i$,

$$\begin{aligned}
&V^*(x,i) - \tilde{V}(x,i) = R^u(x,i,d^u_*(x,i),\pi^l_*[i,d^u_*(x,i)]) - R^u(x,i,\tilde{d}^u(x,i),\tilde{\pi}^l[i,\tilde{d}^u(x,i)]) \\
&\quad + \gamma \sum_{y \in X} \sum_{j \in I} \delta^{\pi^l_*}(x,i,d^u_*(x,i))[y] P^u(j|i,d^u_*(x,i)) V^*(y,j) - \delta^{\tilde{\pi}^l}(x,i,\tilde{d}^u(x,i))[y] P^u(j|i,\tilde{d}^u(x,i)) \tilde{V}(y,j) \\
&\quad \leq \gamma \sum_{y \in X} \sum_{j \in I} \delta^{\tilde{\pi}^l}(x,i,\tilde{d}^u(x,i))[y] P^u(j|i,\tilde{d}^u(x,i)) [V^*(y,j) - \tilde{V}(y,j)] + \gamma\mu \cdot \mathrm{sp}(V^*) \\
&\qquad \text{from the inequality of the previous observation.}
\end{aligned}$$

Now by majorization,

$$\max_{x,i}(V^*(x,i) - \tilde{V}(x,i)) \leq \gamma \cdot \max_{x,i}(V^*(x,i) - \tilde{V}(x,i)) + \gamma\mu \cdot \mathrm{sp}(V^*)$$

Therefore, for all $x$ and $i$ with $0 < \alpha < 1$,

$$0 \leq V^*(x,i) - \tilde{V}(x,i) \leq \frac{\gamma\mu R_{\max}(1-\alpha^T)}{(1-\gamma)(1-\alpha)}.$$

Note that we can define $\tilde{d}^u$ and $\tilde{d}^l$ with respect to a bounded value function $U$ that approximates $V^*$ and draw an error bound from $V^*$ by using the above result we just have drawn.

## 3.3 Heuristic on-line methods

The discussion so far dealt with "off-line" methods for solving MMDPs. Even though various approximation/exact algorithms can be applied for some control problems, it will often require analyzing and utilizing structural properties on the problems, which might be very cumbersome in many interesting problems. In this section, we discuss how to apply previously published some on-line (sampling-based) heuristic techniques in the context of MMDPs.

The first example approach called "(parallel) rollout" is based on the decision rule/policy improvement principle in the "policy iteration" algorithm (see, e.g., [4] [9] [10]). We simulate or rollout a heuristic decision rule available in on-line manner via Monte-Carlo simulation at each decision time and use the estimated value of following the heuristic decision rule by simulation to create an (approximately) improved decision rule with respect to the heuristic decision rule. A generalization of the single decision rule rollout is to rollout in parallel multiple heuristic decision rules and use the maximum estimated value among heuristic decision rules to create a new decision rule. This is particularly useful if system trajectories or sample paths can be divided in a way that a particular heuristic decision rule is near-optimal for particular system trajectories. The parallel rollout method yields a decision rule that dynamically combines the multiple decision rules automatically adapting to different system trajectories and improves the performances of all of the heuristic decision rules. There are several works that reported this (parallel) rollout approach is quite successful. See, e.g., [10] and references therein for the works in this direction.

We briefly discuss how to apply rollout. Suppose we have a heuristic decision rule pair of $d^l$ for the lower level and $d^u$ for the upper level. At each decision time $n$ (in the slow time-scale), we measure the utility of taking each candidate action $\lambda \in \Lambda$ as follows. We take a candidate action (in an imaginary sense) and then from the next step, we simulate $d^l$ and $d^u$ over a finite horizon via Monte-Carlo simulation over many random simulated traces, giving the approximate value of following the decision rule pair. The single step reward of taking action $\lambda$ associated with the lower level quasi-steady state performance is also estimated by simulation by following the decision rule $d^l$. The sum of the estimated single step reward (plus the immediate reward of taking $\lambda$) plus the estimated value of following the decision rule pair $d^l$ and $d^u$ gives the utility measure of the candidate action $\lambda$. At each time $n$, we take the action with the highest utility measure. At the fast time scale, we just follow the decision rule $d^l$.

The Monte-Carlo simulation method is an example for simulation strategy. Depending on problem nature, we can use other simulation methods, e.g., single-path simulation, TD($\lambda$) (see, e.g., [5]), importance sampling, etc. In particular, the single-path simulation for the lower level MDP will be useful if $T$ is relatively large and the underlying Markov chain induced by a decision rule pair is ergodic. In fact, we can draw a probabilistic error bound on the estimate of the value of following a decision rule (or a decision rule pair in our context) from the single path simulation from the true value of the decision rule with respect to the degree of ergodicity and the simulation horizon by using Theorem 2 given in [17].

The above (parallel) rollout approach can be referred as a lower bound approach as the value of following any decision rule pair is a lower bound to the optimal finite-horizon value. The next example called “hindsight optimization” [11] is based on the upper bound. Hindsight optimization can be viewed as a heuristic method of adapting the (deterministic) optimal sample-path based solutions into an on-line solution. Instead of evaluating a decision rule pair by simulation as in the rollout, for each simulated random trace of the system, the optimal control *action sequence* that maximizes the reward sum is obtained. The average over many random traces will give an upper bound on the optimal finite-horizon value. We use this upper bound in the action utility measure. The hindsight optimization approach turns out to be effective in some problems (see, e.g., [8] [43]) even though the question of when this approach is useful is still open. However, we note that as long as the *ranking* of the utility measures of candidate actions reflects well the true ranking (especially the highest one), these heuristic methods can be expected work well.

## 4 Related Works

In this section, we compare several key papers that can be related with our work in hierarchical modeling subject.

We first discuss a key paper by Sutton *et al.* [36] because this paper cites almost all of the hierarchical MDP works in (at least) artificial intelligence literature and some in the control literature and generalizes the previous works by one framework. For many interesting decision problems (e.g., queueing problems), the state spaces in different levels, ($X$ and $I$), are non-overlapping. Sutton *et al.*’s work considers a multi-time MDP model in the dimension of the action space only (action hierarchy) by defining “*options*” or “temporally extended” actions. That is, the state spaces in different time-scales are the same in the Sutton’s model. An option $O$ consists of three components: a stochastic terminating function $g$, the set of states that $O$ can be taken, and a mapping $f$ from state to action. Once an option $O$ is taken from a state, the action from $f$ is taken at each decision time, with possibly terminating via the terminating function. In particular, if $g$ specifies the time duration of applying the function $f$, the option $O$ is called *semi-Markov option.* That is, a sequence of actions in the action space of the given MDP is a temporally extended action or an option. Note that this option *does not* determine or change the underlying reward or state transition structure. On the other hand, in our model, the upper level action $\lambda \in \Lambda$ is not temporally extended action from the action space of the lower level MDPs but is a control at its own right. We can roughly say that the lower level policies defined over different upper level state and controls are semi-Markov options that depend on the upper level state and action.

A similar hierarchical structure in the dimension of only action space was studied in the Markov slowscale model and the delayed slowscale Model by Jacobson *et al.* [20]. They consider two level action hierarchy, where the upper level control is not necessarily an option. However, the upper level control does not change the transition and reward structure of the whole $T$-horizon evolutionary process.

The recent work by Ren and Krogh on multi-mode MDPs [31] studies a nonstationary MDP, where a variable called the system operating mode determines evolution of the MDP, which operates in the one time-scale. We can view our model as a generalization of multi-mode MDPs by viewing each upper level decision and/or state as a system operating mode.

Even though the situation being considered is totally different, Pan and Basar's work [29] considers a class of differential games that exhibit possible multi-time scale separation. Given a problem defined in terms of a singularly perturbed differential equation, differently time-scaled games are identified and each game is solved *independently* and from this a composite solution is developed, which is an approximate solution for the original problem. In our model, the upper level MDP solution must depend on the solution for the lower level MDPs.

Finally, as we mentioned before we can view our model as an MDP-based extension or a generalization of Trivedi's hierarchical performability and dependability model. In Trivedi's work, the performance models (fast time-scale model) are *solved* to obtain performance measures (corresponding to $R^*$ roughly under the lower level policy independent $\delta$-function). These measures are used as reward rates which are assigned to states of the dependability model (slow time-scale). The dependability model is then solved to obtain performability measures. The lower level is modeled by a continuous-time Markov chain and the upper level is modeled by a Markov reward process (alternatively, generalized stochastic petri net. can be used). We can see that if we fix the upper level and the lower level decision rules in our model with the lower level policy independent $\delta$-function, an MMDP becomes (roughly) the model described by Trivedi's — in our model, the lower level model is also a Markov reward/decision process.

## 5 Example Problems

There are many interesting problems that can be modeled by MMDPs. In this section we illustrate this by considering several representative examples in different contexts. The description for each problem is rather abstract but detailed enough to convey the idea of the MMDP modeling. We first discuss a hierarchical extension of the well-known (controlled) multiarmed bandit problem. The next two examples are queueing control problems that arise in communication network. We then consider a hierarchical optimal asset allocation problem that arises in a capital market, a hierarchical production planning problem in a manufacturing environment, and a hierarchical employee staffing problem in a service management environment. We then give an extension example of Trivedi's composite performance and dependability model by incorporating controls into the model. As the final example, a stochastic variant of a deterministic combinatorial graph-based optimization problem is discussed.

## 5.1 Multiarmed bandit problem

It is well-known that multiarmed bandit problem models a class of many interesting (stochastic) optimization problem such as project selection, clinical trials, random search, etc., just to name a few (see, e.g., [40] and the references therein) and the problem can be modeled by MDP. It would be natural to consider a hierarchical version of this problem.

There are $Q$ independent machines. At each time $n$, we need to select exactly one machine to operate based on both of the upper and lower level states of each machine. An upper level state $i_n$ at time $n$ is $(i_n^1, i_n^2, ..., i_n^Q)$ and the lower level state $x_{t_{nT}}$ at time $n$ is $(x_{t_{nT}}^1, x_{t_{nT}}^2, ..., x_{t_{nT}}^Q)$. Once a particular machine is selected (upper level control), say $q$, only the states of the machine $q$ changes and all of the other machines freeze. That is, $i_{n+1}^q$ is determined from a Markovian transition function and starting with $x_{t_{nT}}^q$, the lower level MDP evolves according to the lower level transition function. All states of the other machines remain the same. Note that the lower level transition function will depend on only the upper level *state* of the activated machine. The $T$-epoch reward from the lower level will be incurred to the upper level as one-step reward of operating the machine $q$. The problem is to maximize the expected total discounted sequence of (multi-level) rewards by a decision rule pair of operating machines at each slow time-scale time and of controlling the fast-time scale machine dynamics.

This problem has been solved by Gittins who gave an index rule: under the assumption that $\delta$-function is independent of the lower level policies, straightforward adaptation of the index rule gives an index $G_q$ for machine $q$ such that given the lower level state $x_q$ and the upper level state $i_q$,

$$G_q(x_q, i_q) = \max_{\tau > 1} \frac{E^{x_q, i_q}[\sum_{n=1}^{\tau-1} \gamma^n R_q^*(x_n^q, i_n^q)]}{E^{x_q, i_q}[\sum_{n=1}^{\tau-1} \gamma^t]},$$

where the maximization is over the set of all stopping times $\tau > 1$ and $R_q^*$ is the optimal $T$-horizon value associated with the machine $q$ for the lower level MDP. Then the upper decision rule that operates the machine with the highest index at each time $n$ achieves the two-level optimal infinite horizon discounted value.

Even though this extension is interesting, as we mentioned above, the upper level decision does not affect the lower level MDP dynamics, only the upper level state. Varaiya *et al.* [40] extended the standard bandit problem with an additional freedom: when a particular machine is selected to operate, a control action that affects both the reward and the machine state transition must be also selected and call this extended problem as *controlled* multiarmed bandit problem or "superprocess". The superprocess is naturally modeled by MMDP with the lower level MDP dynamics defined with $P^l(y|x, i, \lambda)$ and $R^l(x, a, i, \lambda)$ as usual and the upper level with $P^u(j|i, \lambda)$ and $R^*$ with the lower level policy independent $\delta$-function. The optimal upper level decision rule for this problem is characterized by the notion of "dominating machine" and the Gittins index rule (see [40] for this matter and applications of the superprocess).

We believe that this example is particularly worthful because it illustrates that there exists a class of problems that can be modeled as MMDPs such that characterization of the optimal upper level decision rule and efficient computation of it are possible.

## 5.2 Admission control with buffer management

Our description of the following example deals with a continuous-time domain. By the *uniformization* method, the continuous-time model can be casted into a discrete-time model (see, e.g., [39]). There are two "call-level" traffic classes. For each class, there is a (real number) weight representing the importance of the class. We can either *accept or reject* of each call arrival. The accepted calls are aggregated into a single (finite) FIFO buffer. At most $N < \infty$ calls can be in the system.

An upper level state $i \in I$ is $(\eta_1, \eta_2)$, where $\eta_c$, $c = 1, 2$, is the number of the pending (currently effective) class-$c$ calls and an upper level action $\lambda \in \Lambda$ is $(\tau_1, \tau_2)$, where $\tau_c \in \{0, -1\}$ and $\tau_c = 0/(-1)$ means accepting/rejecting a newly arrived class-$c$ call. At slow time-scale, class-$c$ calls arrive with an arrival rate $a_c$ according to Poisson process and each (accepted) call's holding time is exponentially distributed with a mean $m_c$ from which we can determine $P^u(j|i, \lambda)$ for $j, i \in I$ and $\lambda \in I$. We want to maximize the weighted number of accepted calls at the upper level.

Given an upper level current state $i$ and the current decision $\lambda$ of rejecting/accepting the new call(s) at the upper level state, the number of effective calls for the next (slow time-scale) epoch will be determined so that the nature of the packet arrival dynamics would be different over each (slow time-scale) epoch. The evolution of the upper level state (after an appropriate uniformization) is given as:

$$i_{n+1} := [\max\{i_n + (\text{new call arrival(s) at } n), N\} + \lambda_n] - \text{ call departure(s) at } (n+1)^-, i_0 = 0,$$

and $[\max\{i_n + (\text{new call arrival(s) at } n), N\} + \lambda_n]$ will determine the packet arrival dynamics over $[t_{nT}, t_{(n+1)T}]$-period at the fast time-scale. The max operator in the above equation is implicitly associated with a discard rule — giving a priority to the more important class for chance of being accepted or rejected when we need to discard calls due to overflow of the buffer. It is assumed that we have a model for each class that describes packet arrival dynamics at the fast time-scale given the number of currently pending calls. The model has a finite number of the *traffic states* and for each traffic state, there is an associated probability distribution such that it gives the probability that $b \in \{b_0, b_1, ..., b_k\}$, $k < \infty$, rate of packets are generated at the traffic state given the number of effective calls. Then the lower level MDP dynamics over each $T$-epoch will be determined through the structure of the model, which includes an appropriate choice of $\delta$-initialization function. We remark that one might want to use a more complex model that can describe the real traffic more suitably. However, the above model suffices to illustrate the usefulness of the MMDP model.

The lower level state $x \in X$ is $(x_1, x_2, s_1, s_2)$, where $x_c$ is the number of the class-$c$ packets (of the same size with the unit time in the fast time-scale) in the buffer, and $s_c$ is the traffic state of the class-$c$, which is observable for simplicity (for the unobservable case, $s_c$ can be a probability distribution over the traffic states called *information state*, in which case the evolution of the lower level states is still Markovian). A lower level action $a \in A$ is $(d_1, d_2)$, where $d_c$ is the number of the class-$c$ packet to be dropped from the tail. The state evolution can be expressed via Lindley's equation involving a particular action for the $x_c$ part and the stochastic transition of $s_c$ part.

We wish to determine the value of $d_c$'s at each (fast time-scale) decision time such that the weighted average queue size (roughly corresponding to average waiting time of packets) is minimized and at the same time the weighted average throughput is maximized, where there is a tradeoff parameter compromising the competence between throughput and queue size/delay. A related buffer management work on a single class problem has been reported in [9], in which motivations for such early packet dropping are given. The lower level reward function is given such that we want to maximize the weighted number of accepted calls and the sum of the weighted (finite-horizon average) throughput and the weighted average queue length with a tradeoff parameter $\xi$:

$$R^l(x, a, i, \lambda) = \frac{1}{T}\left(\sum_c w_c(\mathrm{sgn}(x_c - d_c) \cdot (1 - \xi(x_c - d_c)) + \sum_c w_c(\tau_c + 1)\right),$$

where $w_c$ is the weight of class-$c$ call/packet.

## 5.3 Call routing with buffer management

The next example problem is a simple call-routing problem with buffer management and motivated from a "load-balancing" control problem that arises in for example, traffic engineering at MPLS (Multi-Protocol Label Switching) domain [7].

Consider a network with $M$ parallel links (called label switched path in MPLS domain), $m = 1, ..., M$ between two points of source and destination. At the source, single class (voice) calls arrive with an arrival rate according to Poisson process in a slow time-scale. The call's holding time is exponentially distributed with a mean in the slow time-scale. Again, we can use the uniformization method for continuous-time arrival/departure process. The call-level or upper level decision process is either to reject a newly arrived call or to decide where to dispatch or route the newly arrived call to one of the $M$ parallel links if accepted. We assume that for each link, there are (possibly zero) cross traffic (video) calls. For simplicity, the video calls are initially set up and do not depart (if we incorporate the dynamics of video call arrival and departure process, we would have a three-level MMDP and the control process in the highest level is to assign video calls among $M$ parallel links or reject).

It is assumed that all of the voice calls have the same traffic rate (i.e., bandwidth requirement) and this is also true of the video calls. In other words, the model describing packet (of the same size with the unit time in fast time-scale) arrival process of the voice call is the same. For example, if On/Off model is used, each arriving call has the same On/Off model parameters. This also holds for the video calls. We may use, e.g., Markov modulated Poisson process to model video packet arrival process [12] and it is also assumed that each video call shares the same model parameters. We then can induce a model like the one we discussed in the previous example section for voice/video traffic that describes packet arrival dynamics at the fast time-scale given the number of currently pending voice/video calls with an appropriate choice of the $\delta$-initialization function at each link.

The upper level state consists of the number of currently pending voice and video calls at each link. The lower level state consists of the traffic states for voice traffic and video traffic and the number of packets in the (finite FIFO) buffer for voice and video at each link. The control action at the lower level is to drop voice and video packets from the tail as in the previous example problem at each link. The lower level reward function is given by the sum of the reward function at the individual link plus reward of routing/rejecting a voice call.

We remark that as a variant of the above two examples discussed so far, we can consider a scheduling problem instead of the buffer management problem.

## 5.4 Asset allocation

It is important that capital market brokers make proper decisions on shifting their investments to more promising assets depending on market dynamics. As the next example problem for MMDP, we consider a very simple asset allocation (portfolio management) problem that deals with the investment of capital to various trading opportunities (e.g., different stocks). The example here is based on the problem discussed in [28] and appropriately modified into our contexts.

We assume that there is a set of market states ($I$ in the upper level) that triggers the "government" to change or stick to its decision rule ($\Lambda$ in the upper level), which affects the trend of the exchange rate in the capital market. A transition from a market state to another state would be a function of the government decision rule and some exogenous random disturbances. A possible trend of the exchange rate follows an increasing pattern, but with higher values of the exchange rate, a drop to very low values becomes more and more probable [28].

A lower level state $x$ consists of the current exchange rate $e$, the capital (the wealth of the portfolio) $c$, which is always calculated in the basis currency, and a variable, representing which currency (e.g., US dollars) is currently used for the investment. The stochastic nature of the exchange rate will determine the lower level MDP dynamics. The action at the lower level to be taken by a broker is to choose a proper currency for the investments, to maximize the expected return for a given finite-horizon, (if a lower level policy independent $\delta$-function is used), where the return is a function of the exchange rate, the capital, and the transaction costs. The optimal portfolio composition will depend on a government decision rule and a market state, and the government needs to select a decision rule based on the effects of the investments of the brokers in the market to maximize the overall return.

## 5.5 Production planning

Hierarchical production planning problems have been studied in operations research literature with certain models and assumptions over many years (see, e.g., [33] for references). In this subsection, we give a simple production planning problem in a manufacturing environment as the next example of the MMDP model. We base our discussion on the problem studied in [6].

The production planning problem we consider here is divided into two levels: "marketing management" level and "operational" level. At the marketing management level, we need to control which *family* to produce over each (slow time-scale) decision epoch, where a family is a set of items consuming the same amount of resources and sharing the same setup [6]. The upper level state consists of (stochastically) available resources for each family and (stochastic) setup costs for each family, and some market-dependent factors. The upper level action is to choose which family to produce.

At the operational level, we need to determine actual quantities of the items in the family (the lower level actions) given stochastic (Markovian) demands for the items, production capacity, holding cost, material cost, etc., which will constitute a state of the lower level.

The return at the operational level will be a function of the unit selling price of the items, the inventory holding costs, the setup costs of the (current) family, the production quantity of the items, etc. and the $T$-epoch expected accumulated return at the operational level will be the one-step return for the management level.

## 5.6 Employee staffing

The next example mirrors in some sense the problem in a manufacturing environment we just discussed in the previous section but gives a good insight again about the usefulness of the MMDP model and is based on the work in [44].

Employee staffing at a service organization company (e.g., supermarket, telephone call center, etc.) is a very important management problem because how to approach to this problem is directly related with the revenue of the company.

At the level-1, we have "employ hiring" problem. Suppose the company categorizes the employees into certain types depending on different skills and service capacities at different purposes. The state at the level-1 is the currently hired number of employees for each type. This number will change at random in a monthly basis due to for example, downsizing, company job-training, employee's leaving, etc. and we assume that we have a Markovian model to describe this phenomenon (see, e.g., [45] for this direction). The control at the level-1 is to hire how many new employees for a certain type or not to hire.

Given the currently effective working number of employees at the company, depending on desired activity volume (stochastic customer demands), work duration, (stochastic) company budget, corporate rules, etc., we need to dispatch the right number of the employees for each type to the right place in a weekly basis. This will define the level-2 MDP problem often called "workforce scheduling" problem.

At the lowest level-3 (a fast time-scale), we need to solve a real-time control problem. For example, at a telephone call center, we need to assign incoming calls, which will be stochastic, to available customer service representatives.

## 5.7 Composite (controlled) performance and dependability model

In this subsection, we give an extension of the example discussed by Goševa-Popstojanova and Trivedi [15] to show how we can use MMDP to extend their hierarchical model of performance and dependability (or availability) into an interesting model that incorporates controls. We don't base our discussion on the exactly same example given in [15]. Our example is appropriately modified from the example into our contexts.

There are $N$ multiprocessors, where each processor has a different service characteristic and a different failure rate. The concept of "dependability" is related with failure, reconfiguration, and repair of these processors with respect to system behavior, and closely related with dependability, availability measures a potential service capacity that the system can deliver [15]. For our extension, we naturally introduce a "maintenance" operation in a slow time-scale. We wish to maintain the system such that a better availability/dependability is achieved. The measure of availability will depend on the performance made by each processor in a fast time-scale.

Each processor $k$'s upper level state has two parts. The first part indicates whether the processor is up or down. The second part is the processor $k$'s currently effective exponential service rate $S$ in a finite set of $\{S_{k,1}, ..., S_{k,m}\}$ if the first part is up, where we assume that there is a Markov process that describes transition of service rates for each processor $k$ in the slow time-scale. If the first part is down, the second part indicates a current "symptom" or reasons of failure from a finite set of $\{F_{k,1}, ..., F_{k,m}\}$. The processor $k$ is subject to failure $F_{k,n}$ with an exponential rate $f_{k,n}$. An upper level action is to select one processor to be repaired among currently failed processors. The immediate repair cost at the upper level is, for example, a time to repair associated with symptom $F$.

At the lower level, jobs or customers of type $k$ (associated with weight $w_k$) arrive at processor $k$ according to Poisson process with an arrival rate of $a_k$. Each job requires a processing time that is exponentially distributed with a rate $\tau_k$ and it is assumed that on a job's arrival the processing time is known to the processor. The lower level state is the queue size at each processor $k$ in terms of the total processing time for the jobs in the queue. The lower level action for each processor is to do admission control to newly arriving jobs based on the current processor's service rate (and its future stochastic variation if we use the lower level dependent $\delta$-function) to effectively control the throughput and the waiting time of the processed jobs based on a given tradeoff parameter. The performance made by each processor $k$ times $w_k$ will contribute to the upper level single step reward.

Our overall goal in this example is to maintain the system (at both levels) to make overall system's availability optimal over infinite horizon. In addition to our example here, many interesting variants of Trivedi *et al.*'s composite performance and availability model can be considered via the MMDP modeling.

## 5.8 Salesman's travel planning

As the final example, we consider a simple but somewhat artificial stochastic variant of a deterministic *graph-based combinatorial optimization problem.* Consider a salesman who needs to travel from a source city to a destination city by a long load trip. We assume that there is a connected network of possible routes of intermediate cities between the source and the destination.

The upper level problem is just a variant of the shortest path problem. An upper level state is a city and an upper level action is the choice of the next city to visit. The upper level state transition will be deterministic in this case and the destination city will be the termination state in the upper level MDP such that once entered, the salesman stays forever with the zero cost.

The salesman needs to visit small towns along the trip between a pair of cities. Between two towns, he needs to select a method of transportation (taxi, bus, train, or walk, etc.) depending on his current budget, stochastic load condition due to the weather for example, and so forth. It is assumed that the hotel, food, etc. prices are randomly changing too according to a Markovian model so that he needs to consider this random factor in his decision time-scale when he makes a decision. The cost will be a function of the money he spends and the travel time by the selected transportation. This will determine the lower level MDP dynamics. The budget plan will affect the decision at the upper level to choose the next city to visit and the goal is to minimize the overall cost of traveling from the source city to the destination city.

We remark that the problem above can be made more interesting if we incorporate "multi-time scale" budget condition into the upper level MDP dynamics so that a particular city must be excluded from the travel plan (a random topological change in the graph). There are many interesting stochastic variants of deterministic graph-based combinatorial optimization problems modeled as stochastic shortest path problems, which consist of a subclass of the MDP model [3] [5] (see, e.g., vehicle routing problems in [32] and references therein). Usually, those graph-based problems are associated with certain weight functions in the arcs in the graphs. We can consider lower level MDP problems that replace those weight functions giving rise to MMDP problems.

## 6 Concluding Remarks

We proposed in this paper an analytical model called multi-time scale MDP for a class of hierarchically structured sequential decision making processes. The model describes interactions between levels in the hierarchy in a pyramid sense. The upper level state and/or control induces the lower level MDP dynamics over a finite horizon of length corresponding to the relevant scale factor. The performance measure obtained from the lower level will affect the upper level decision making.

A hierarchical objective functions for this model was devised and corresponding optimality equations were established, from which for each objective function, we could derive the property of optimal value functions and the existence of optimal policies at all levels. We then presented the exact and approximate solution methods and also discussed heuristic (on-line) methods to solve MMDPs.

In the evolutionary process of MDPs, the outcome of taking an action at a state is the next state. Usually, the matter of *when* this outcome is known to the system is not critical as long as the system comes to know the next state before the next decision time. However, this might be an issue on the MMDP model. In our model, we assumed that the next state at the upper level is known at the near boundary of the next time step (refer Figure 1), which is quite reasonable we believe. If the effect of taking an action $\lambda \in \Lambda$ at a state $i \in I$ is immediate, which is the next state $j \in I$, $P^l(y|x,i,\lambda)$ will be possibly given as $P^l(y|x,j)$. This issue is the problem specific matter and needs to be resolved by the system design.

We made the assumption that action spaces at all levels in the hierarchy are distinct. Even though we believe that this is a natural assumption, we speculate that for some applications, some actions might be shared by different levels. Our assumption can be relaxed (with added complexity to the model) so that some actions are shared by different levels as long as any action taken at a state in a level does not affect the higher level state transitions. Developing a model for the case where a lower level action affects the higher level transitions is still open problem.

The extension of our model into *partially observable* MMDP is straightforward because a partially observable MDP can be transformed into an MDP with information state space (see, e.g., [1]). Furthermore, it will be interesting to extend our model into the Markov game settings making multi-time scaled Markov games. The "optimal equilibrium value of game" over a finite horizon at lower level game will be used as one-step cost/reward for the upper level game. We believe that the MMDP model we proposed has many applications, in particular in the areas of stochastic queueing control and production planning.

## References

[1] A. Arapostathis, V. S. Borkar, E. Fernández-Gaucherand, M. K. Ghosh, and S. I. Marcus, "Discrete-time controlled Markov processes with average cost criterion: a survey," *SIAM J. Control and Optimization*, vol. 31, no. 2, pp. 282–344, 1993.

[2] T. Basar and G. J. Olsder, *Dynamic Noncooperative Game Theory*, Academic Press, London/New York, 1995.

[3] D. P. Bertsekas, *Dynamic Programming and Optimal Control, Volumes 1 and 2.* Athena Scientific, 1995.

[4] D. P. Bertsekas and D. A. Castanon, "Rollout algorithms for stochastic scheduling problems," *J. of Heuristics,* vol. 5, pp. 89–108, 1999.

[5] D. P. Bertsekas and J. Tsitsiklis, *Neuro-Dynamic Programming.* Athena Scientific, 1996.

[6] G. Bitran, E. A. Hass, and K. Matsuo, "Production planning of style goods with high set-up costs and forecast revisions," *Oper. Research*, vol. 34, pp. 226–236, 1986.

[7] R. Callon, P. Doolan, N. Feldman, A. Fredette, G. Swallow, and A. Viswanathan, "A framework for multiprotocol label switching," Internet draft <draft-ietf-mpls-framework-05.txt>, Sep. 1999.

[8] H. S. Chang, R. Givan, and E. K. P. Chong, "On-line scheduling via sampling," in *Proc. 5th Int. Conf. on Artificial Intelligence Planning and Scheduling* , 2000, pp. 62–71.

[9] H. S. Chang, R. Givan, and E. K. P. Chong, "Model-based random early packet dropping using rollout policies," submitted to *Discrete Event Dynamic Systems: Theory and Application*, 2000.

[10] H. S. Chang and S. I. Marcus, "On approximate receding horizon control for Markov decision processes: average reward case," submitted to *Automatica* (TR 2001-46, ISR, Univ. of Maryland, 2001).

[11] E. K. P. Chong, R. Givan, and H. S. Chang, "A framework for simulation-based network control via hindsight optimization," in *Proc. 39th IEEE CDC*, 2000, pp. 1433–1438.

[12] W. Fischer and K. Meier-Hellstern, "The Markov-modulated Poisson process (MMPP) cookbook," *Performance Evaluation*, vol. 18, pp. 149–171, 1992.

[13] J. Forestier and P. Varaiya, "Multilayer control of large Markov chains," *IEEE Trans. on Automatic Control,* vol. AC-23, No. 2, pp. 298–304, 1978.

[14] S. B. Gershwin, "Hierarchical flow control: a framework for scheduling and planning discrete events in manufacturing systems," *Proc. of the IEEE*, vol. 77, no. 1, pp. 195–208, 1989.

[15] K. Goseva-Popstojanova and K. S. Trivedi, "Stochastic modeling formalisms for dependability, performance and performability," *Performance Evaluation - Origins and Directions, Lecture Notes in Computer Science*, G. Haring, C. Lindemann, M. Reiser (eds.), pp. 385–404, Springer Verlag, 2000.

[16] M. Grossglauser and D. Tse, "A time-scale decomposition approach to measurement-based admission control," submitted to *IEEE/ACM Trans. on Net.*

[17] P. Glynn and D. Ormoneit, "Hoeffding's inequality for uniformly ergodic Markov chains," *Statistics and Probability Letters,* 2001. To appear.

[18] O. Hernández-Lerma, *Adaptive Markov Control Processes.* Springer-Verlag, 1989.

[19] O. Hernández-Lerma and J. B. Lasserre, "Error bounds for rolling horizon policies in discrete-time Markov control processes," *IEEE Trans. on Automatic Control*, vol. 35, no. 10, pp. 1118–1124, 1990.

[20] M. Jacobson, N. Shimkin and A. Shwartz, "Piecewise stationary Markov Decision Processes, I: constant gain," submitted to *Mathematics of Operations Research.*

[21] M. Kearns, Y. Mansour, and A. Y. Ng, "A sparse sampling algorithm for near-optimal planning in large Markov decision processes," in *Proc. 16th International Joint Conf. on Artificial Intelligence,* 1999, pp. 1224–1231.

[22] M. Mahmoud, "Multilevel systems control and applications: a survey," *IEEE Trans. on Systems, Man, and Cybernetics,* vol. SMC-7, No. 3, pp. 125–143, 1977.

[23] M. Malhotra and K. Trivedi, "A methodology for formal expression of hierarchy in model solution," in *Proc. of the 5th Int. Workshop on Petri Nets and Performance Models*, 1993, pp. 258–267.

[24] M. Mandjes and A. Weiss, "Sample path large deviations of a multiple time-scale queueing model," submitted, 1999.

[25] J. Muppala, M. Malhotra, and K. Trivedi, "Markov dependability models of complex systems: analysis techniques," *Reliability and Maintenance of Complex Systems*, S. Ozekici (ed.), pp. 442–486, Springer-Verlag, Berlin, 1996.

[26] S. Meyn, "The policy iteration algorithm for average reward Markov decision processes with general state space," *IEEE Trans. on Automatic Control*, vol. 42, no. 12, pp. 1663–1680, 1997.

[27] A. Müller, "How does the value function of a Markov decision process depend on the transition probabilities?," *Math. of Operations Research*, vol. 22, no. 4, pp. 872–885, 1997.

[28] R. Neuneier, "Optimal asset allocation using adaptive dynamic programming," in *NIPS'95*, D. Touretzky, M. Mozer, M. Hasselmo (eds), MIT Press, 1996.

[29] Z. Pan and T. Basar, "Multi-time scale zero-sum differential games with perfect state measurements," *Dynamics and Control*, vol. 5, pp. 7–30, 1995.

[30] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming.* Wiley, New York, 1994.

[31] Z. Ren and B. H. Krogh, "Mode-matching control policies for multi-mode Markov decision processes," in *Proc. of ACC*, vol. 1, 2001, pp. 95–100.

[32] N. Secomandi, "Comparing neuro-dynamic programming algorithms for the vehicle routing problem with stochastic demands," *Computers and Operations Research*, vol. 27, pp. 1201–1225, 2000.

[33] S. P. Sethi and Q. Zhang, *Hierarchical Decision Making in Stochastic Manufacturing Systems*, in series Systems and Control: Foundations and Applications, Birkhäuser Boston, Cambridge, MA, 1994.

[34] A. Shwartz and A. Weiss, "Multiple time scales in Markovian ATM models I. Formal calculations," CC PUB 267, Electrical Engineering, Technion, 1999.

[35] S. Singh and R. Yee, "An upper bound on the loss from approximate optimal-value functions," *Machine Learning*, vol. 16, pp. 227–233, 1994.

[36] R. Sutton, D. Precup, and S. Singh, "Between MDPs and Semi-MDPs: a framework for temporal abstraction in reinforcement learning," *Artificial Intelligence*, vol. 112, pp. 181–211, 1999.

[37] D. Tse, R.G. Gallager and J.N. Tsitsiklis, "Statistical multiplexing of multiple time-scale Markov streams," *IEEE J. on Selected Areas in Communications*, vol. 13, no. 6, pp. 1028–1039, 1995.

[38] T. Tuan and K. Park, "Multiple time scale congestion control for self-similar network traffic," *Performance Evaluation*, vol. 36-37, pp. 359–386, 1999.

[39] R. Serfozo, "An equivalence between continuous and discrete time Markov decision processes," *Operations Research*, vol. 27, pp. 616–620, 1979.

[40] P. Varaiya, J. Walrand, and C. Buyukkoc, "Extensions of the multiarmed bandit problem: the discounted case," *IEEE Trans. on Automatic Control*, vol. AC-30, no.5, pp. 426–439, 1985.

[41] W. Willinger, M. Taqqu, W. Leland and D. Wilson, "Selfsimilarity in high-speed packet traffic: analysis and modeling of Ethernet traffic measurements," *Stat. Sci.* vol. 10, pp. 67–85, 1995.

[42] W. Willinger, M. Taqqu, and A. Erramilli, "A bibliographical guide to self-similar traffic and performance modeling for modern high-speed networks," in *Stochastic Networks: Theory and Applications*, F.P. Kelly, S. Zachary, and I. Ziedins (ed.), Royal Statistical Society Lecture Note Series 4, Clarendon Press, Oxford, pp. 339–366, 1996.

[43] G. Wu, E. K. P. Chong, and R. Givan, "Congestion control via online sampling," in *Proc. of INFOCOM*, 2001, pp. 1271–1280.

[44] Y. Zhou and N. Gans, "Managing learning and turnover in employee staffing," To appear in Operations Research.

[45] Y. Zhou and N. Gans, "A single-server queue with Markov-modulated service times," submitted for publication, 1999.